

Tesina de Grado  
para la obtención del título de  
Licenciado en Ciencias de la Computación

# Seguridad en iOS y Android: un Análisis Comparativo

---

Autor

**Raúl Ignacio Galuppo**

raul.i.galuppo@gmail.com

G-3483/5

Director

*Dr. Carlos Luna*



Departamento de Ciencias de la Computación  
Facultad de Ciencias Exactas, Ingeniería y Agrimensura  
Universidad Nacional de Rosario  
Diciembre de 2017

# Índice general

<b>1. Hacia un Framework Comparativo</b>	<b>1</b>
1.1. Vista principal . . . . .	1
1.1.1. Funciones no compatibles . . . . .	2
1.2. Catálogo de test . . . . .	3
1.2.1. Contactos . . . . .	3
1.2.2. Calendario . . . . .	4
1.2.3. Geolocalización . . . . .	5
1.2.4. SMS . . . . .	7
1.2.5. Almacenamiento . . . . .	8
1.2.6. DeviceInfo . . . . .	8
1.2.7. Sensores . . . . .	9
1.2.8. Internet . . . . .	10
1.3. Resultados experimentales . . . . .	11
1.3.1. Clase A . . . . .	12
1.3.2. Clase B . . . . .	13
1.3.3. Clase C . . . . .	13

# Índice de figuras

1.1. Áreas de la aplicación <i>Runtime Permissions Test</i> . . . . .	2
1.2. Testeando la administración de los contactos. . . . .	4
1.3. Testeando la administración del calendario. . . . .	5
1.4. Testeando la geolocalización. . . . .	6
1.5. Panel de configuración de coordenadas del emulador de Android. . . . .	6
1.6. Testeando los mensajes SMS. . . . .	7
1.7. Testeando el almacenamiento del dispositivo. . . . .	8
1.8. Obteniendo datos del dispositivo. . . . .	9
1.9. Testeando los sensores. . . . .	10
1.10. Testeando el acceso a Internet. . . . .	11

# Capítulo 1

## Hacia un Framework Comparativo

Android e iOS permiten cambiar ciertos permisos de una aplicación luego de haberla instalado en el dispositivo. En este contexto, se ha desarrollado un framework para determinar empíricamente el alcance de dichos cambios en los sistemas de permisos de ambas plataformas.

El framework es una aplicación móvil y está compuesto por varios tests. Cada test pone a prueba a un componente del dispositivo, permitiendo así conocer el alcance de los permisos correspondientes a dicho componente. De esta manera, se busca dejar en evidencia posibles vulnerabilidades presentes en los modelos de seguridad. Adicionalmente, se hace especial enfoque a la relación existente entre la privacidad del usuario y el sistema de permisos, analizando cuál es la cobertura del sistema respecto de los datos sensibles para la privacidad.

En las siguientes secciones se detallarán los distintos tests que componen el framework. Además, se mencionarán las conclusiones arribadas luego de correr los tests mencionados anteriormente.

### 1.1. Vista principal

Al iniciar la aplicación, se observan dos áreas principales: Acciones y Test, como se puede observar en la Figura 1.1.

La primera área contiene un botón para acceder a la configuración de los permisos del dispositivo. Allí, el *tester* puede cambiar manualmente los permisos requeridos por la aplicación. Además, se encuentra un botón para limpiar la consola del framework.

La segunda área se subdivide en dos: la parte de los tests y la parte de la consola. Una parte corresponde a los botones de los tests que, al presionarse, ejecutan el respectivo test, mostrando en la consola el resultado. Dicho resultado se mostrara con tipografía color verde si fue exitoso; en cambio,

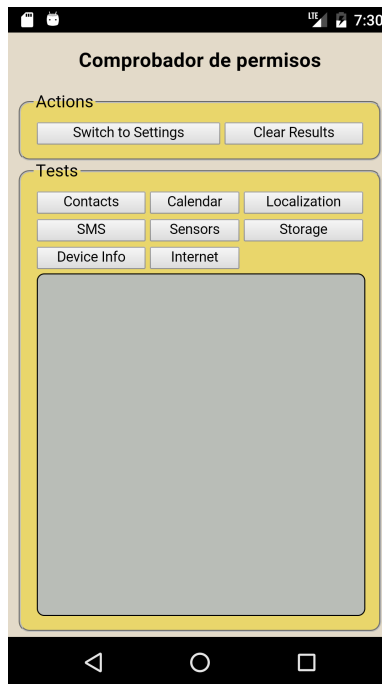


Figura 1.1: Áreas de la aplicación *Runtime Permissions Test*.

se mostrara con tipografía color roja de ser fallido.

A continuación se mencionan los componentes que se pueden testear con el framework:

- Contactos
- Calendario
- Geolocalización
- SMS
- Sensores
- Almacenamiento
- Información del dispositivo.
- Acceso a Internet

#### 1.1.1. Funciones no compatibles

El emulador oficial de Android es compatible con la mayoría de las funciones de un dispositivo, pero no incluye la posibilidad de virtualizar los siguientes componentes [3]:

- WiFi;
- Bluetooth;
- NFC<sup>1</sup>;
- Manipulación de la tarjeta SD;
- Conexión USB;
- Micrófono;
- Cámara

Al no poder manipular la tarjeta SD, no es posible testear las funcionalidades multimedia: no se puede grabar audio, ni video ni sacar fotos.

Por lo tanto, no se agregaron al framework tests para los componentes listados anteriormente.

## 1.2. Catálogo de test

En esta sección se listarán todos los test que conforman al framework. Para cada test se detalla su algoritmo, los plugins de Apache Cordova que se utilizaron para desarrollarlo y una serie de capturas que muestran los casos exitosos y fallidos.

Para acceder al panel de configuraciones, se utilizó el siguiente plugin: `cordova.plugins.diagnostic v3.1.7`

### 1.2.1. Contactos

---

#### **Algorithm 1** Test de Contactos.

---

- 1: Se listan todos los contactos en la consola.
  - 2: Se crea un nuevo contacto.
  - 3: Se vuelven a listar todos los contactos en la consola.
- 

El test consiste en crear un contacto y luego listar todos los contactos presentes en el dispositivo. En caso de ser exitoso, se muestran los contactos. De lo contrario, se muestra un error. En la Figura 1.2a se observa el resultado del test cuando no se tiene el permiso correspondiente; mientras que en la Figura 1.2b se observa el caso exitoso.

Para desarrollarlo, se utilizó el plugin `cordova-plugin-contacts` (v. 2.2.1) de Apache Cordova.

Finalmente, para correr el test, es necesario tener el permiso **Contacto**, tanto para Android como para iOS.

---

<sup>1</sup>Del ingles *Near Field Communication*. Es una tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia que permite el intercambio de datos entre dispositivos.



(a) Sin permisos.

(b) Con permisos.

Figura 1.2: Testeando la administración de los contactos.

### 1.2.2. Calendario

---

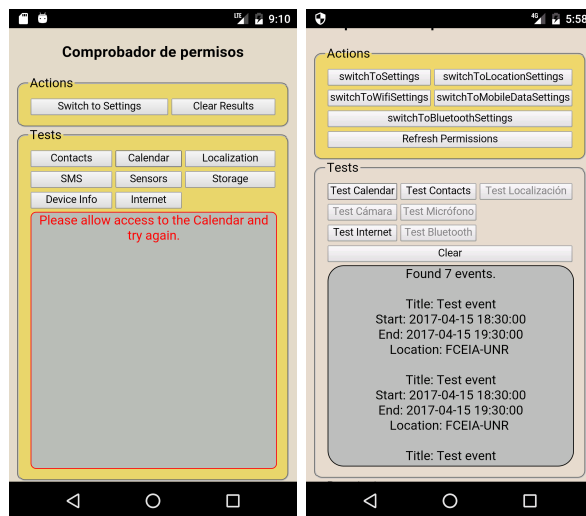
#### Algorithm 2 Test del Calendario.

---

- 1: Se crean las fechas *startDate* y *endDate*.
  - 2: Se crea un evento que empieza en la fecha *startDate* y termina en la fecha *endDate*.
  - 3: Se listan en la consola los eventos entre las fechas *startDate* y *endDate*.
- 

El test consiste en crear un evento en un determinado rango de fechas y luego listar todos los eventos dentro del rango. En caso de ser exitoso, se muestran los datos del evento. De lo contrario, se muestra un error. En la Figura 1.3a se observa el resultado del test cuando no se tiene el permiso correspondiente; mientras que en la Figura 1.3b se observa el caso exitoso. Para desarrollarlo, se utilizó el plugin `cordova-plugin-calendar` (v. 4.5.5) de Apache Cordova.

Finalmente, para correr el test, es necesario tener el permiso `Calendario` para Android. En cambio, para iOS, es necesario tener el permiso `Recordatorios`.



(a) Sin permisos.

(b) Con permisos.

Figura 1.3: Testeando la administración del calendario.

### 1.2.3. Geolocalización

---

#### **Algorithm 3** Test de Geolocalización.

---

- 1: Se censa el GPS.
  - 2: Se muestran los datos en la consola.
- 

El objetivo del presente test es obtener los datos de la ubicación actual. En caso de tener los permisos correspondientes, obtiene tanto la ubicación precisa (GPS) como la aproximada (WIFI/Móvil). En la Figura 1.4a se observa el resultado del test cuando no se tiene el permiso correspondiente; mientras que en la Figura 1.4b se observa el caso exitoso.

Para desarrollarlo, se utilizó el plugin cordova-plugin-geolocation (v. 2.4.3) de Apache Cordova.

Al momento de realizar las pruebas, se configuró el emulador de Android para que simule las coordenadas ( $-122^\circ$ ,  $37^\circ$ ), tal como se observa en la Figura 1.5. Dicha configuración también se realizó en emulador oficial de iOS.

Finalmente, para correr el test, es necesario tener el permiso **Localización**, tanto para Android como para iOS.





(a) Sin permisos.

(b) Con permisos.

Figura 1.4: Testeando la geolocalización.

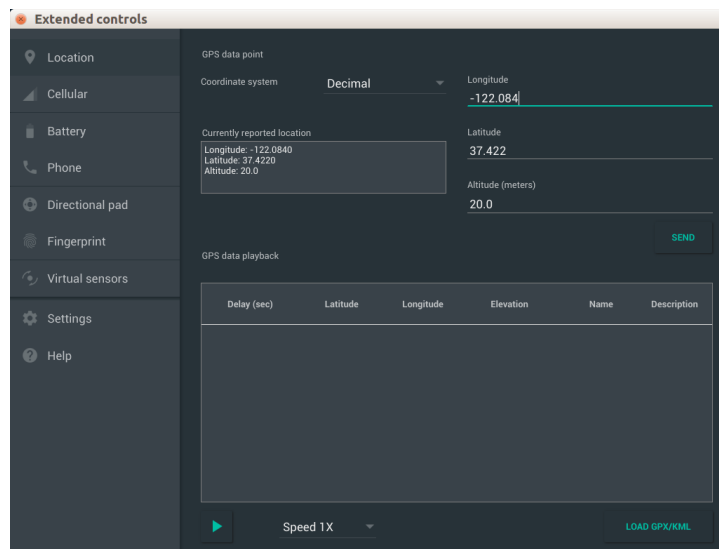


Figura 1.5: Panel de configuración de coordenadas del emulador de Android.

#### 1.2.4. SMS

---

**Algorithm 4** Test de SMS.

---

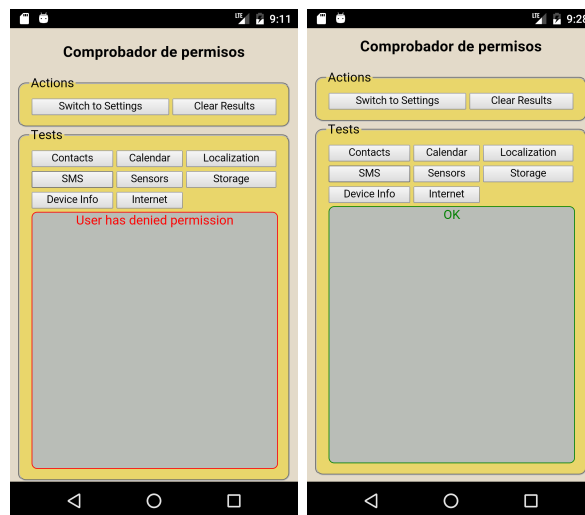
- 1: Se inicializan los eventos para recibir SMS.
  - 2: Se envía un SMS de prueba.
  - 3: Se indica el resultado del test en la consola.
- 

El test consiste en enviar un mensaje SMS. En un principio, se diseñó con la capacidad de enviar, leer y recibir mensajes. Al momento de desarrollarlo, se encontró una restricción de seguridad en iOS: a partir de la version 8 no se pueden acceder a los mensajes SMS desde una aplicación instalada por el usuario [5, 6]. En cambio, en Android sí se pueden acceder, siempre que se tengan el permiso correspondiente. Dicho permiso es **SMS**.

Como consecuencia de lo mencionado en el párrafo anterior, se decidió no implementar las funcionalidades incompatibles, quedando en el test la posibilidad de enviar mensajes SMS. Para desarrollarlo, se utilizó el plugin cordova-plugin-sms (v. 1.0) de Apache Cordova.

En la Figura 1.6a se observa el resultado del test cuando no se tiene el permiso correspondiente; mientras que en la Figura 1.6b se observa el caso exitoso.

Finalmente, para correr el test, es necesario tener el permiso **SMS** para Android. Sin embargo, no es necesario tener permisos para correr el test en iOS.



(a) Sin permisos.

(b) Con permisos.

Figura 1.6: Testeando los mensajes SMS.

### 1.2.5. Almacenamiento

---

**Algorithm 5** Test de Almacenamiento.

---

- 1: Se intenta capturar la pantalla.
  - 2: Se guarda la captura en el dispositivo.
- 

El presente test fue diseñado para probar el alcance de los permisos de escritura sobre el sistema de archivos que tiene cada plataforma. En la Figura 1.7a se observa el resultado del test cuando no se tiene el permiso correspondiente; mientras que en la Figura 1.7b se observa el caso exitoso. Para desarrollar el presente test se utilizó el plugin cordova-screenshot. Finalmente, para correr el test, es necesario tener el permiso **Almacenamiento** para Android. Sin embargo, no es necesario tener permisos para correr el test en iOS.

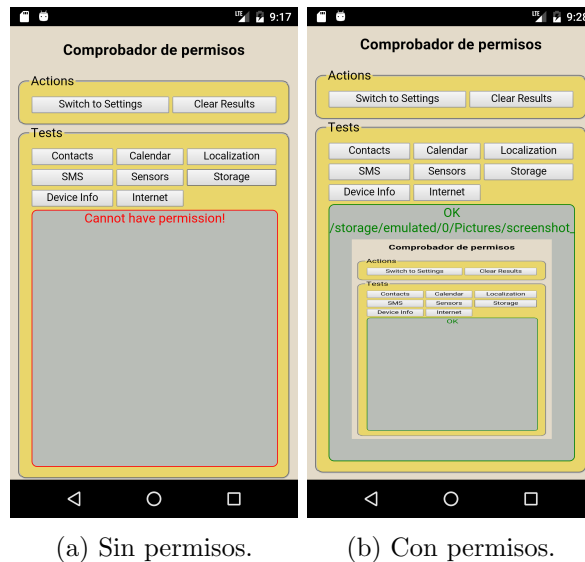


Figura 1.7: Testeando el almacenamiento del dispositivo.

### 1.2.6. DeviceInfo

---

**Algorithm 6** Test de Información del Dispositivo.

---

- 1: Se obtienen los datos del dispositivo.
  - 2: Se imprime por consola los datos obtenidos.
- 

El objetivo de este test es obtener datos del dispositivo donde corre la aplicación. Entre otros datos, se obtiene la plataforma, modelo del dispositivo y su número de serie. En la Figura 1.8 se observan los datos

obtenidos.

Para desarrollarlo se utilizó el plugin cordova-plugin-device.

Cabe aclarar que no fueron necesarios permisos en ninguna de las dos plataformas para poder correrlo.

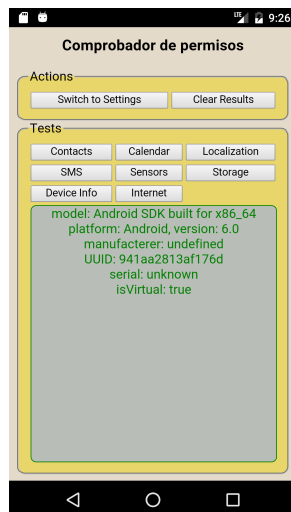


Figura 1.8: Obteniendo datos del dispositivo.

### 1.2.7. Sensores

---

**Algorithm 7** Test de los Sensores.

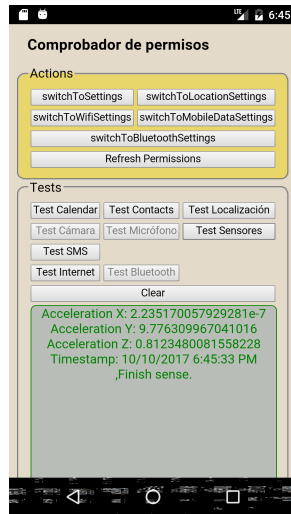
---

- 1: Se inicializa un **timer** con 5 **seg** para detener las mediciones.
  - 2: Se inicia la medición del acelerómetro.
  - 3: Se inicia la medición del giroscopio.
  - 4: Se muestran los resultados en la consola.
- 

El objetivo de este test es obtener datos de dos sensores del dispositivo: acelerómetro y giroscopio. Para ello, se configura un **timer**. Durante el tiempo que este activo, se tomarán distintos muestreos; y cuando ocurra el *timeout* se mostrarán los datos en la consola de la aplicación. En la Figura 1.9a se observan los datos obtenidos.

Para desarrollar el presente test se utilizaron los tests cordova-plugin-device-motion y cordova-plugin-gyroscope, de Apache Cordova.

Cabe aclarar que no fueron necesarios permisos en ninguna de las dos plataformas para poder correrlo.



(a) Datos obtenidos.

Figura 1.9: Testeando los sensores.

### 1.2.8. Internet

---

#### **Algorithm 8** Test de conexión a Internet.

---

- 1: Se realiza una consulta GET HTTP hacia logo del DCC
  - 2: Se decodifica la imagen (viene codificada en Base64).
  - 3: Se muestra en consola un `tag <IMG>`, cuyo `source` es el dato decodificado.
  - 4: Se realiza una consulta POST HTTP hacia `httpbin`
  - 5: Se muestra en consola la respuesta.
- 

El objetivo del presente test es establecer una comunicación a través de Internet. No se requirió de ningún plugin para desarrollarlo. En la Figura 1.10a se observa el resultado del test cuando no se tiene conexión a Internet; mientras que en la Figura 1.10b se observa el caso de establecer una comunicación por Internet.

Cabe aclarar que, al ejecutarse el framework desde un emulador, para probar el acceso a Internet se habilitó/deshabilitó la Red Inalámbrica de la computadora donde se corrieron los emuladores.

Por último, para correr el presente test, no fueron necesarios permisos en ninguna de las dos plataformas.



(a) Sin conexión a Internet. (b) Con conexión a Internet.

Figura 1.10: Testeando el acceso a Internet.

### 1.3. Resultados experimentales

El framework desarrollado en este trabajo tiene la capacidad de testear los componentes *Contactos*, *Calendario*, *Geolocalización*, *SMS*, *Sensores*, *Almacenamiento*, *Información del dispositivo* y *Acceso a Internet*, tal como se explica en la sección 1.1.

Un primer resultado es poder clasificar los componentes testeados en tres clases, según requieran explícitamente permisos para utilizarlos. Dichas clases son:

- Clase A: componentes que requieren autorización explícita en ambas plataformas para poder utilizar sus funcionalidades;
- Clase B: componentes que requieren autorización explícita solamente en Android;
- Clase C: componentes que no requieren permisos para poder utilizar sus funcionalidades.
- Clase D: componentes que requieren autorización explícita solamente en iOS;

Las clases son mutuamente excluyentes. Luego de correr los tests, se confeccionó la Tabla 1.1. Cabe aclarar que la *Clase D* es puramente teórica, ya que ninguno de los componentes analizados cayeron en dicha clase. Sin

embargo, es bastante razonable que exista algún componente en dicha clase, que pueda ser descubierto al ser ampliado el framework.

<i>Permisos</i>			
<i>Clase A</i>	<i>Clase B</i>	<i>Clase C</i>	<i>Clase D</i>
Contactos	-	-	-
Calendario	-	-	-
Geolocalización	-	-	-
-	SMS <sup>2</sup>	-	-
-	Almacenamiento	-	-
-	-	Sensores	-
-	-	Información del dispositivo	-
-	-	Acceso a Internet	-

Cuadro 1.1: Clasificación de permisos según si requieren autorización.

### 1.3.1. Clase A

Los componentes que conforman esta clase son *Contactos*, *Calendario* y *Geolocalización*. La característica común entre ellos es que requieren autorización explícita del usuario para interactuar con ellos. Si el usuario deniega el correspondiente permiso, no se puede acceder a ninguna funcionalidad del componente.

El primer componente a analizar es *Contactos*. En Android, requiere el permiso *peligroso* llamado Contacto y en iOS requiere el permiso llamado con el mismo nombre. En ambas plataformas, los dos permisos abarcan las mismas funcionales: permiten crear un contacto, borrarlo, editarlo y obtener un listado de todos los contactos presentes en el dispositivo.

Luego, se continua analizando el componente *Calendario*. Permite administrar todo lo relacionado con los eventos: crear un evento, modificarlo y eliminarlo. Además, permite administrar varios calendarios. Cada evento está relacionado a un calendario. En iOS, estas funcionalidades están separadas en dos componentes: *Calendarios* y *Recordatorios*. Cada uno de ellos tiene su permiso que lo regula. Sin embargo, el test solo prueba el uso del permiso llamado Recordatorios. Por otra parte, en Android el permiso *peligroso* que regula estas funcionalidades se llama Calendario.

A continuación se analiza el último componente de esta clase, llamado *Geolocalización*. En ambas plataformas, provee el acceso a la ubicación del dispositivo, ya sea la ubicación precisa (GPS) o la aproximada (WIFI/Móvil). Tanto en Android como en iOS, el permiso que regula dichas

<sup>2</sup>Aplica solamente al envío de mensajes.

funcionalidades se llama Localización.

### 1.3.2. Clase B

Esta clase está compuesta por los componentes *SMS* y *Almacenamiento*. Ellos tienen en común que para utilizar sus funcionalidades no requieren permisos en iOS. Sin embargo, necesitan autorización explícita en Android. Se comenzará el análisis por el componente *SMS*. Sus funcionalidades son: enviar un mensaje, eliminarlo, obtener los mensajes entrantes y obtener la lista de todos los mensajes, ya sean recibidos o enviados. Sin embargo, el test que se desarrolló solamente prueba el envío de mensajes, ya que en iOS, a partir de la versión 8, no se permite acceder al resto de las funcionalidades desde una aplicación de terceros (es decir, no nativa) [5, 6]. En cambio, Android permite acceder a todas las funcionalidades mencionadas; y se pueden acceder a través del permiso *peligroso* llamado SMS. Finalmente, se hablará sobre el componente *Almacenamiento*. En Android, el permiso que regula al componente mencionado tiene su mismo nombre. Dicho permiso resguarda las funcionalidades que permiten leer y escribir la tarjeta SD. En el test, se intenta escribir en el sistema de archivos.

### 1.3.3. Clase C

El objetivo de esta clase es agrupar componentes que tienen permisos *normales* en Android. A pesar de ello, pueden ser potencialmente peligrosos a la hora de ser utilizados para violar la privacidad de un usuario *TODO: EXPLICAR!*. De todos los componentes que tienen relacionados algún permiso *normal*, se eligieron tres: *Información del dispositivo*, *Sensores* y *Acceso a Internet*. Cabe aclarar que ambos no requieren autorización explícita del usuario, en ninguna de las dos plataformas, para utilizar las funcionalidades que brindan.

El primer componente nos brinda datos relacionados al dispositivo en el cual se está corriendo la aplicación. Especifica el modelo del dispositivo, el cual es establecido por el fabricante del dispositivo y puede ser diferente en todas las versiones del mismo producto. También se puede obtener información relacionada a la plataforma del dispositivo: nombre, versión, quién lo manufacturó, su UUID<sup>3</sup> y si es emulado o no.

Luego, se continúa con el componente *Sensores*. Dicho componente controla el acceso a los sensores del dispositivo. El test recolecta datos de dos sensores: el acelerómetro y el giroscopio.

El último componente de esta clase es el que nos provee acceso a Internet. En la actualidad, son muchas las aplicaciones móviles que utilizan Internet.

---

<sup>3</sup>Del inglés Universally Unique Identifier. Es un número de 128 bits que identifica al dispositivo biunívocamente.



Según el ranking confeccionado por [10], las 10 aplicaciones más populares utilizan Internet. El test permite realizar una petición GET y una petición POST sin notificar al usuario. Esto es potencialmente muy peligroso ya que una aplicación maliciosa puede enviar y recibir datos sin que el usuario lo note. *TODO: dar más detalles de esto!*

# Bibliografía

- [1] ANDROID. *Android Open Source Project: Security*. <https://source.android.com/security/index.html>. Último acceso 4 de Noviembre del 2017.
- [2] ANDROID. *Android Security 2015 Year In Review*. [https://source.android.com/security/reports/Google\\_Android\\_Security\\_2015\\_Report\\_Final.pdf](https://source.android.com/security/reports/Google_Android_Security_2015_Report_Final.pdf), 2016. Último acceso 4 de Noviembre del 2017.
- [3] ANDROID, D. *Developer Android: Funciones no compatibles*. <https://developer.android.com/studio/run/emulator.html#about>.
- [4] APPLE. *Apple iOS: Security Guide*. [https://www.apple.com/business/docs/iOS\\_Security\\_Guide.pdf](https://www.apple.com/business/docs/iOS_Security_Guide.pdf), March 2017. Último acceso 4 de Noviembre del 2017.
- [5] APPLE, F. D. *Forum Developer Apple: How to listen for sms reception in ios 8?* <https://forums.developer.apple.com/thread/16685>. Último acceso 4 de Noviembre del 2017.
- [6] APPLE, F. D. *Forum Developer Apple: How to read user's sms in an application?* <https://forums.developer.apple.com/thread/22447>. Último acceso 4 de Noviembre del 2017.
- [7] BETARTE, G., CAMPO, J. D., LUNA, C. D., AND ROMANO, A. Verifying android's permission model. In *ICTAC 2015* (2015), LNCS 9399, pp. 485–504.
- [8] GOROSTIAGA, F. Especificación e implementación de un prototipo certificado del sistema de permisos de android. Tesina de grado, Licenciatura en Ciencias de la Computación, Universidad Nacional de Rosario, Argentina, Octubre 2016.
- [9] HAN, J., YAN, Q., GAO, D., ZHOU, J., AND DENG, H. R. Comparing mobile privacy protection through cross-platform applications. *Network & Distributed System Security Symposium (NDSS)* (2013).

- [10] OF APPS, B. *App Download and Usage Statistics 2017*. <http://www.businessofapps.com/data/app-statistics/>. Último acceso 4 de Noviembre del 2017.
- [11] ROMANO, A. Descripción y análisis formal del modelo de seguridad de android. Tesina de grado, Licenciatura en Ciencias de la Computación, Universidad Nacional de Rosario, Argentina, Junio 2014.
- [12] YOGITA CHITTORIA, N. A. Application security in android-os vs ios. *International Journal of Advanced Research in Computer Science and Software Engineering* 4 (2014).